Compression of Textured Surfaces Represented as Surfel Sets

T. Darom [a] M. R. Ruggeri [b] D. Saupe [b] N. Kiryati [a]

[a] *School of Electrical Engineering*
Tel Aviv University, Tel Aviv, Israel

[b] *Dept. of Computer and Information Science*
University of Konstanz, Konstanz, Germany

Abstract

A method for lossy compression of genus-0 surfaces is presented. Geometry, texture and other surface attributes are incorporated in a unified manner. The input surfaces are represented by surfels (surface elements), i.e., by a set of disks with attributes. Each surfel, with its attribute vector, is optimally mapped onto a sphere in the sense of geodesic distance preservation. The resulting spherical vector-valued function is resampled. Its components are decorrelated by the Karhunen-Loève transform, represented by spherical wavelets and encoded using the zerotree algorithm. Methods for geodesic distance computation on surfel-based surfaces are considered. A novel efficient approach to dense surface flattening/mapping, using rectangular distance matrices, is employed. The distance between each surfel and a set of key-surfels is optimally preserved, leading to greatly improved resolution and eliminating the need for interpolation, that complicates and slows down existing surface unfolding methods. Experimental surfel-based surface compression results demonstrate successful compression at very low bit rates.

Key words: textured surface compression, spherical mapping, geodesic paths, surfels, spherical wavelets

⋆ This research was supported by the Kurt Lion Foundation. At Tel-Aviv University, it was supported by the Ministry of Science. At Konstanz University, it was supported by the DFG Graduiertenkolleg "Explorative Analysis and Visualization of Large Information Spaces".

1 Introduction

Various modern image processing applications are inherently three dimensional. Textured three dimensional models are of common use in video games, virtual shops, computer-aided design, telemedicine and medical image databases. This research is concerned with the processing of textured 3D surfaces, in particular with their representation and compression.

Textured 3D models can be either synthetic or real. Synthetic models are generated by 3D modeling or computer-aided design techniques. Real models are obtained by 3D scanners, such as laser or structured light range finders. They can also be obtained by segmentation of 3D medical images, acquired using CT, MRI or other imaging modalities.

Real time transmission and compact storage of detailed 3D models require compression of textured 3D models. For example, many modern video games consist of large virtual worlds stored on servers, that transmit relevant three-dimensional models to the player side client for visualization. Telemedicine applications and efficient indexing in large 3D medical image databases also call for convenient representation of 3D models.

Three dimensional models appear in various formats. Triangle meshes are common in computer graphics; voxel representations are dominant in medical imaging. For triangle-mesh surfaces, compression of both the geometry and texture was considered in [1]. Recently, Pfister *et al* [2] developed a surface representation scheme based on *surfels* (surface elements). Surfels are disks with associated attributes. A surfel s_i can be characterized by its center location p_i, normal vector $\vec{n}_i$, color 3-tuple (r_i, g_i, b_i) and radius ρ_i. Additional properties, such as shininess, opacity and texture characteristics can also be attributed to each surfel. In this representation, a surfel can be viewed as an extended point sample, but without any connectivity implied between surfels. Surface representation by surfel sets is gaining popularity, and is a natural output format for important classes of 3D scanners (laser and structured light, shape from shading and texture, photometric stereo, tactile). This research focuses on the compression and representation of textured surfaces represented by surfel sets.

Geometry compression schemes applicable to surfel-based surfaces were presented by Fleishman *et al* [3] and by Ochotta and Saupe [4]. The latter method consists of division of the surfaces into patches and compression of the patch height field above a suitable reference plane. Compression of both the geometry and texture of point-set surfaces was considered by Waschbüsch *et al* [5]. In their scheme based on multiresolution predictive encoding, geometry and surface attributes such as texture are separately compressed.

We suggest a compression scheme for surfaces represented by surfel sets. Geometry, texture and other surface attributes are represented in a unified framework. The proposed algorithm relies on the observation that in many applications, notably in the important task of 3D face representation, the surface can be mapped to the sphere as a whole. This is convenient for indexing in 3D surface databases.

Lossy surface compression schemes introduce a geometry error. To allow quantitative performance evaluation, an error estimation procedure is needed. For triangle mesh compression, the Metro tool [6] has become the de-facto standard for geometry error estimation. To evaluate the geometry error due to compression of surfaces represented by surfels, we follow the error measurement procedure used by Ochotta and Saupe [4]. In that procedure, the geometry error is measured via the Moving Least Squares surface (MLS surface) [7] representation of the surfel-based surfaces.

Spherical mapping, i.e., the mapping of a given surface onto a sphere, is a major building block in the suggested surface representation and compression scheme. Spherical mapping has various statistical interpretations and applications [8]. Recently, it has been used for face recognition [9] and texture mapping [10].

The input to spherical mapping procedures is generally a matrix quantifying the dissimilarity between points in the dataset. In geometric surface analysis applications [9–11] dissimilarity is measured by geodesic distance. The geodesic distances should be preserved, as much as possible, by the spherical mapping algorithm.

The spherical multi dimensional scaling (Spherical MDS) algorithm, that was presented by Cox and Cox [8], was designed for optimal preservation of dissimilarity values by *Euclidean* distances between points on the sphere. Elad and Kimmel [12] extended the algorithm for optimal preservation of the *geodesic* distances between points on an input surface by distances between the respective points on the sphere.

Memory requirements limit the dimension of the input dissimilarity matrix in [8,12]. In surface analysis applications, this implies that the surface must be down-sampled prior to the spherical MDS procedure. Spherical mapping of the full-resolution input surface can then be accomplished only by interpolation, that cannot generally recover the down-sampling loss.

High quality representation and compression of textured surfaces via spherical mapping requires mapping of the input surface elements at the full resolution. The algorithms of [8,12] are therefore inadequate for our purpose. As part of this research, we developed a computationally viable *dense* solution to the spherical mapping problem.

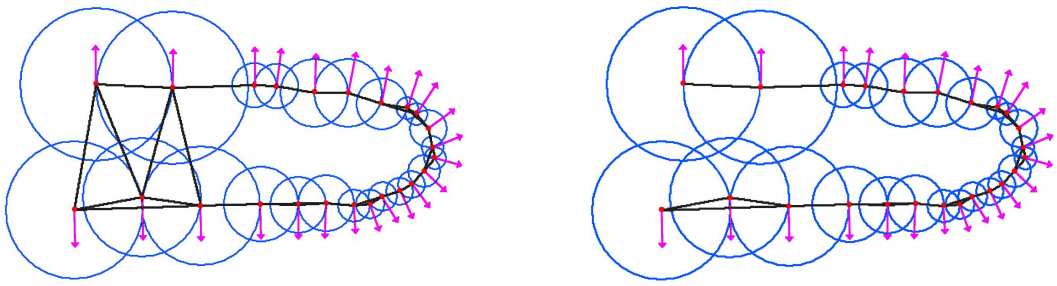
The computation of geodesic distance, i.e., the length of the shortest path between points on the input surface, is central to the suggested approach. An early algorithm for geodesic distance estimation on voxel-based surfaces was presented by Kiryati and Székely [13]. An efficient solution for triangle meshes was developed by Kimmel and Sethian [14] and Surazhsky*et al* [15]. Algorithms for geodesic distance estimation over point-based surfaces were presented by Mémoli and G. Sapiro [16], Hofer and Pottmann [17], and by Klein and Zachmann [18]. In this research, the input surface is represented by surfel sets. An extension of a point based geodesic distance estimation algorithm to the case of surfel based surfaces has therefore been necessary.

The organization of this paper is as follows. Geodesic distance estimation on surfaces represented by surfels is considered in section 2. A computationally efficient procedure for geodesic-distance preserving spherical MDS is presented in section 3. The compression scheme is described in section 4. Experimental results are presented in section 5, followed by discussion.

2 Computing geodesic distances on surfel-based surfaces

The geodesic distance between two arbitrary points p_1 and p_2 on a surface is defined as the length of the shortest path between them on the surface. Geodesics for *point clouds* have been considered before by several authors. Memoli and Sapiro in [16] construct an offset band of the point set surface consisting of the union of Euclidean balls centered at the given points. Geodesics are then computed by applying a Fast Marching algorithm [19] on a 3D Cartesian grid built inside the constructed band. The accuracy depends on the resolution of the 3D grid, and the execution time depends on the number of grid points. Hofer and Pottmann [17] proposed to compute geodesic curves as energy minimizing discrete curves constrained on a moving-least-squares (MLS) surface. They proposed an efficient method for high dimensional minimization of energy functionals on a constraint manifold, which is designed to yield geodesic curves. Klein and Zachmann [18] approximated geodesics as shortest paths on a geometric proximity graph over point clouds. They experimented with Delaunay graphs and spheres-of-influence graphs (SIG) as proximity graphs and proposed an extended version of SIG. Our focus is on the computation of geodesic distances on surfaces represented by *surfel sets*.

As will be seen in section 3, estimates of geodesic distances are needed in this research in order to reduce the parameterization distortion. Specifically, it is necessary to compute the geodesic distances between each of n reference surfels to all N surfels. Thus, we need a fast and efficient algorithm to estimate geodesic distances between surfels over all the surfel-based model. This problem is similar to the well-known problem of finding shortest paths



(a) (b)

Fig. 1. (a) Canonical SIG, and (b) SIG extended with surfel orientations.

from one source to all the nodes of a weighted graph. For our application we construct a weighted graph, which describes surfel connectivity. Among the various graphs proposed in the literature [20] for point samples, we chose to adapt the spheres-of-influence graph (SIG) [18]. It accommodates variations in the point sample density and produces a good description of non-smooth surfaces [20,18].

The sphere centered at point sample (surfel position) v_i with radius given by the distance to the nearest neighbor is called the sphere of influence of v_i. The SIG is the graph with vertices v_i in which two vertices v_i and v_j, $i \neq j$, are connected by an edge e_{ij} if the corresponding spheres of influence intersect. Each edge e_{ij} is weighted by the Euclidean distance between the connected vertices.

As shown in [18], in case of noisy or irregularly sampled point clouds, a SIG could produce small clusters of connected points that are inter-cluster disconnected even though they are part of the same surface. To overcome this problem, different approaches that focus on modifying the radii of the spheres of influence can be applied. In [18] the radius of the sphere centered at v_i is set to the distance to the k-th nearest neighbor with $k > 1$, a parameter of the method. This may produce extraneous long edges in the graph requiring an additional edge pruning step. The surfel-based representation is different from plain point clouds, because surfels are oriented and have a size attribute. We use these attributes to extend our SIG. We set the radius of the sphere at v_i to the radius of the surfel s_i. In fact, the radius of a surfel disk identifies the contribution of its sample point in representing the underlying surface. In addition, we require that two surfels connected by an edge have approximately the same orientation. Using this approach undesirable edges will be discarded, see Fig. 1. We use surfel orientations, i.e., their normal vectors as pre-pruning criterion. We connect two vertices by an edge only if the dihedral angle between their surfel normal vectors is less than a certain threshold angle θ_t (in our implementation, θ_t is set to be slightly larger than a right angle).

The geodesic distance estimate between surfels, that will be needed in the

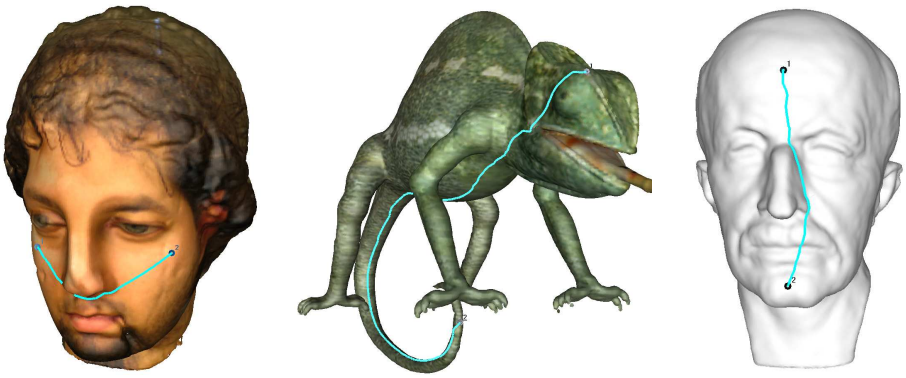


Fig. 2. Geodesic paths (cyan) on three surfel based models.

Table 1
RMS of the relative errors of approximated geodesic distances on a plane, a sphere, and a cylinder. The plane is characterized by a planar square sampled regularly on a grid. The model of the sphere has been obtained by sampling regularly the unit sphere with 4098 surfels. The model of the cylinder has been obtained by sampling a cylinder with diameter being equal to its height, using an equiangular sampling on its circumference and an equidistant sampling on its axis.

Model	# surf.	RMS error
Plane	2500	0.0584
Sphere	4098	0.0148
Cylinder	5002	0.0593

spherical parameterization step of the representation scheme proposed in this paper, can be computed by applying Dijkstra's algorithm on the weighted graph. If necessary, the geodesic path can also be constructed: Fig. 2 shows examples of the geodesic path between selected points on surfel-based surfaces.

This approach provides fast approximation of the geodesic distances on surfel-based surfaces. Our geodesic distance estimates are not precise, since the shortest paths on the graph computed with the Dijkstra algorithm must pass through surfel centers. This results in an approximation error. To evaluate the performance, we considered surfel-based models of surfaces for which the exact geodesic distance can be computed analytically: planes, spheres, and cylinders. We then measured the relative errors between our geodesic distance estimates and the exact geodesic distances. Table 1 reports the root mean square of the relative errors (RMS) computed on different surfel-based surfaces.

The complexity of computing our geodesic distance estimate between a given surfel and all other surfels in the model using Dijkstra's algorithm is $O(|E| + |V| \log |V|)$, where V and E are the sets of surfels (vertices) and

Table 2
Execution time for computing weighted graphs

Model	# surfels	Time (sec.)
Bunny [22]	35,948	3.50
Max Planck [23]	52,809	3.90
Chameleon [23]	101,685	4.73
Igea [23]	134,345	6.08
Lion [24]	183,408	7.16
Imperia(528K) [25]	263,908	17,25
Dragon [23]	1,279,519	56.39

edges, respectively [21]. This procedure must be executed n times, where n is the number of reference surfels. Performances can be improved by using techniques based on the multi-level-bucketing data structure [21], where the average time complexity is linear for uniformly distributed edge lengths. Furthermore, the worst case complexity is $O(|E| + |V| \log |C|)$, where C denotes the ratio between the length of the largest edge and the length of shortest edge.

The computation of the graph takes $O(|V| \log |V|)$ time [20]. Table 2 shows the execution time for computing our extended SIG on surfel-based models with different sizes. The algorithm implemented with non-optimized C++ code has been executed on an Intel Pentium 4 2.80 GHz with 2GB of RAM running under a Windows XP Professional system.

3 Spherical mapping

3.1 Background

Spherical mapping is the mapping of points from a surface to a sphere. Praun and Hoppe [26] mapped vertices of a triangle mesh sequentially, minimizing a local stretch measure. Global relaxation steps at preset intervals complement the process. This method was later used as part of a compression scheme for surfaces represented as triangle meshes [27]. Zwicker and Gotsman [28] considered spherical mapping of a point set by constructing a k-nearest neighbor graph and iteratively reducing the corresponding Laplace-Beltrami operator on the sphere. We use a generalization of the spherical mapping algorithm that was originally developed by Cox and Cox [8], and later extended and

applied to surface analysis by Elad and Kimmel [12].

Cox and Cox [8] define a stress functional

$$S_C = \left[\frac{\sum_{j=1}^{N} \sum_{i=1}^{N} (d_{ij} - \delta_{ij})^2}{\sum_{j=1}^{N} \sum_{i=1}^{n} \delta_{ij}^2} \right]^{\frac{1}{2}} \quad (1)$$

where δ_{ij} is the dissimilarity between data points i and j, d_{ij} is the Euclidean distance between their images on the sphere, and N is the total number of points. Following earlier work on surface flattening via multidimensional scaling [29,30], Elad and Kimmel [12] used (normalized) geodesic distance on the input surface as the dissimilarity measure. They have further redefined d_{ij} as the (normalized) arc length on the destination sphere. In this paper, the data points i and j are surfels. The functional thus quantifies the discrepancy between corresponding geodesic distances on the input surface and on the sphere.

Computing and minimizing this functional requires $\Theta(N^2)$ storage of the dissimilarity matrix. For the processing of practical surfaces, that may contain millions of points, a pre-processing step of surface simplification to at most a few thousands of points has been necessary. In the context of surfaces represented by triangle meshes, Elad and Kimmel [12] minimize the functional using the conjugate gradient algorithm.

3.2 Dense spherical mapping

Surface simplification by sub-sampling (decimation) is incompatible with the preservation of details. *Dense* spherical mapping, i.e., the mapping of *all* the surface points onto the sphere, without interpolation, is therefore necessary. For surface flattening onto the plane, Bankirer and Kiryati [31] suggested relying on the dissimilarity between a small set of n reference points to *all* N points. This is accomplished by measuring the geodesic distance from each reference point to all points on the surface at the *full* resolution. The resulting non-square $n \times N$ dissimilarity matrix is used as input to a modified classical scaling algorithm [31] (compare with [32]). Thus, rather than mapping the $N - n$ non-reference points by interpolation of the mapping of the n reference points, their mapping is tightly determined by their geodesic distances to all reference points. These distances are obtained without additional computational cost, because the geodesic distances from each reference point to the non-reference points is obtained anyway while computing the geodesic distances from the reference point to the other reference points. In the context of surface flattening onto the plane, this has led to substantial gains, in terms of both accuracy and computation time.

Here, the concept of [31] is adapted to the spherical domain. A subset of n points (surfels) out of all the N points are evenly selected using the furthest point strategy [33]. The resulting $n \times N$ dissimilarity matrix is used to define a modified stress functional:

$$S_D = \frac{\sum_{i=1}^{n} \sum_{j=1}^{N} (d_{ij} - \delta_{ij})^2}{\sum_{i_1}^{n} \sum_{j=1}^{N} \delta_{ij}^2} \tag{2}$$

The functional is minimized in five steps:

(1) Given the $n << N$ reference points (surfels), they are mapped to the sphere by minimizing functional (1) using the conjugate gradients algorithm [34].
(2) The mapping of all N points is obtained by interpolation [35], and used to initialize the next step.
(3) The functional (2) is minimized by moving only non-reference points, using the conjugate gradients method.
(4) A relaxation step takes place, in which the functional is minimized using the conjugate gradients method.
(5) The solution is rotated by 90^o along the meridian, and the functional is minimized again, without moving points that are close to the sphere poles.

Steps 3-4, and the minimization in step 5, can be iterated. The fifth stage is necessary, because the numerical solution is unstable near the poles. In principle, this problem can be traced to [8] and [12], but is more noticeable here, because of the density of the points near the poles. By minimizing this functional, each point (surfel) is mapped onto the sphere, eliminating the need for surface simplification, sub-sampling or interpolation.

3.3 Performance evaluation

In order to quantitatively and visually demonstrate the performance gains made possible by the *dense* spherical mapping approach, a mapping error is defined for each surfel:

$$\epsilon_i = \left[\sum_{\forall s_j \in S_r} \frac{(d_{ij} - \delta_{ij})^2}{{\delta_{ij}}^2} \right]^{\frac{1}{2}} \tag{3}$$

where S_r is a set of 100 randomly selected surfels.

Table 3 shows the global mapping errors (RMS values of the surfel mapping errors) obtained using the dense spherical mapping method presented in this

Table 3
Global mapping quality with the suggested dense spherical mapping method

reference surfels (n)	RMS mapping error
10	0.1334
50	0.1326
100	0.1311

Table 4
Global mapping quality using constrained spherical MDS [12] with interpolation

reference surfels (n)	RMS mapping error
1000	0.1459
2000	0.1453
5000	0.1444

paper, for typical numbers of reference points n. Table 4 is provided as reference, and presents the corresponding performance of the algorithm of [12] followed by interpolation. Note that the number n of reference points needed to achieve a certain error level using the suggested method is smaller by about two orders of magnitude with respect to the method of [12]. The respective color mappings of the error are shown in figures 3 and 4. The mapping accuracy gain made possible by the proposed method is evident.

4 Textured surface compression

The compression of surfaces that are represented by triangle meshes has received significant attention, see e.g. [36,37,1]. In some cases, the preservation of the mesh connectivity is an important consideration [36]. In others, remeshing takes place as part of the compression-reconstruction cycle [1]. In surfel-based surface representation there is no mesh, hence there is no connectivity that needs to be preserved. Nevertheless, in analogy to the remeshing notion, the surface can be "resurfeled" as part of the process, meaning the approximate representation of the surface by a new surfel set. We focus on lossy compression of surfel-based surfaces, that maintains good visual quality at high compression ratios.

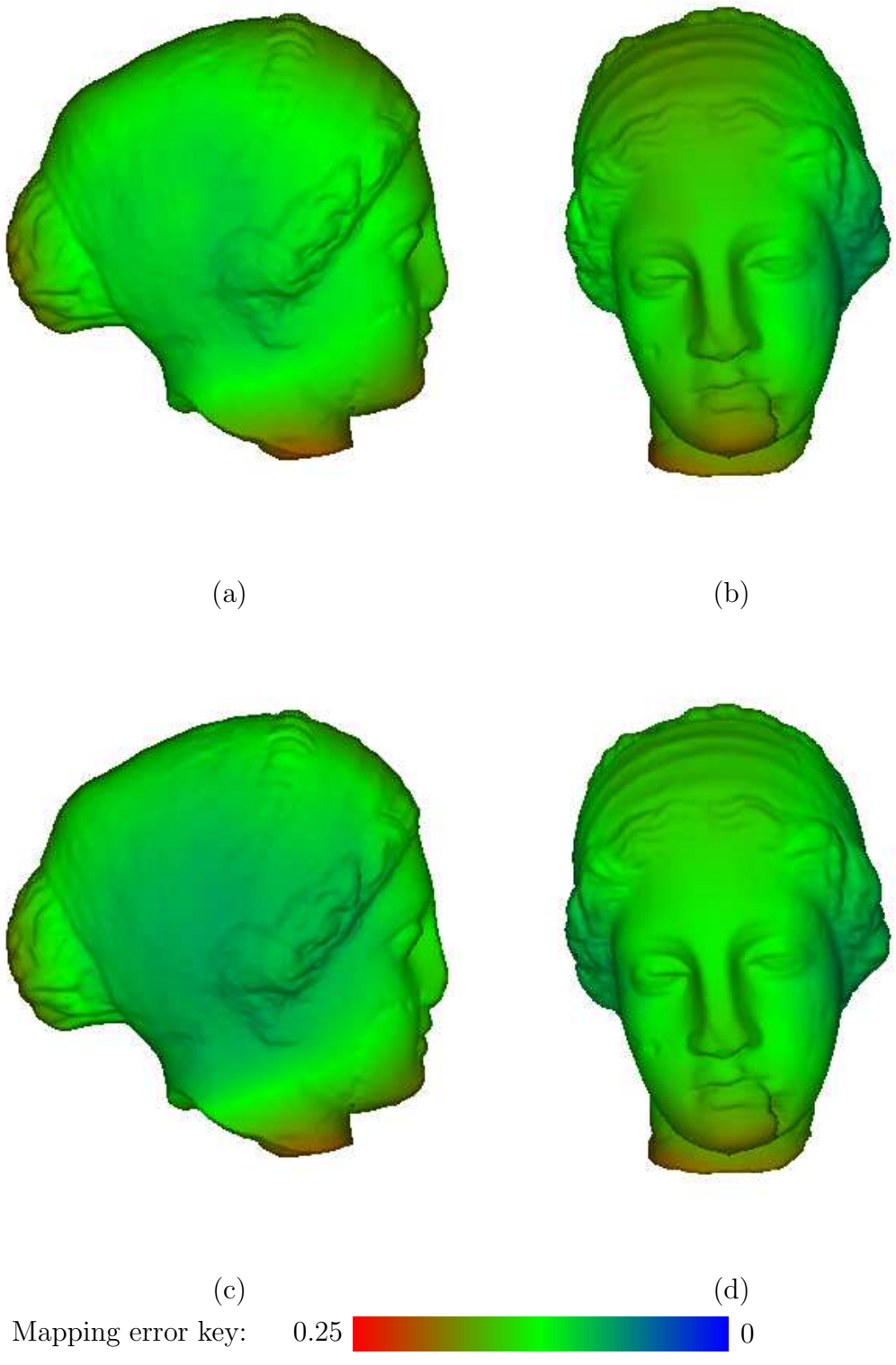


Fig. 3. Dense spherical MDS: The mapping error of each surfel in the Igea model is color-coded. (a), (b): Using 10 reference surfels. (c), (d): Using 150 reference surfels.

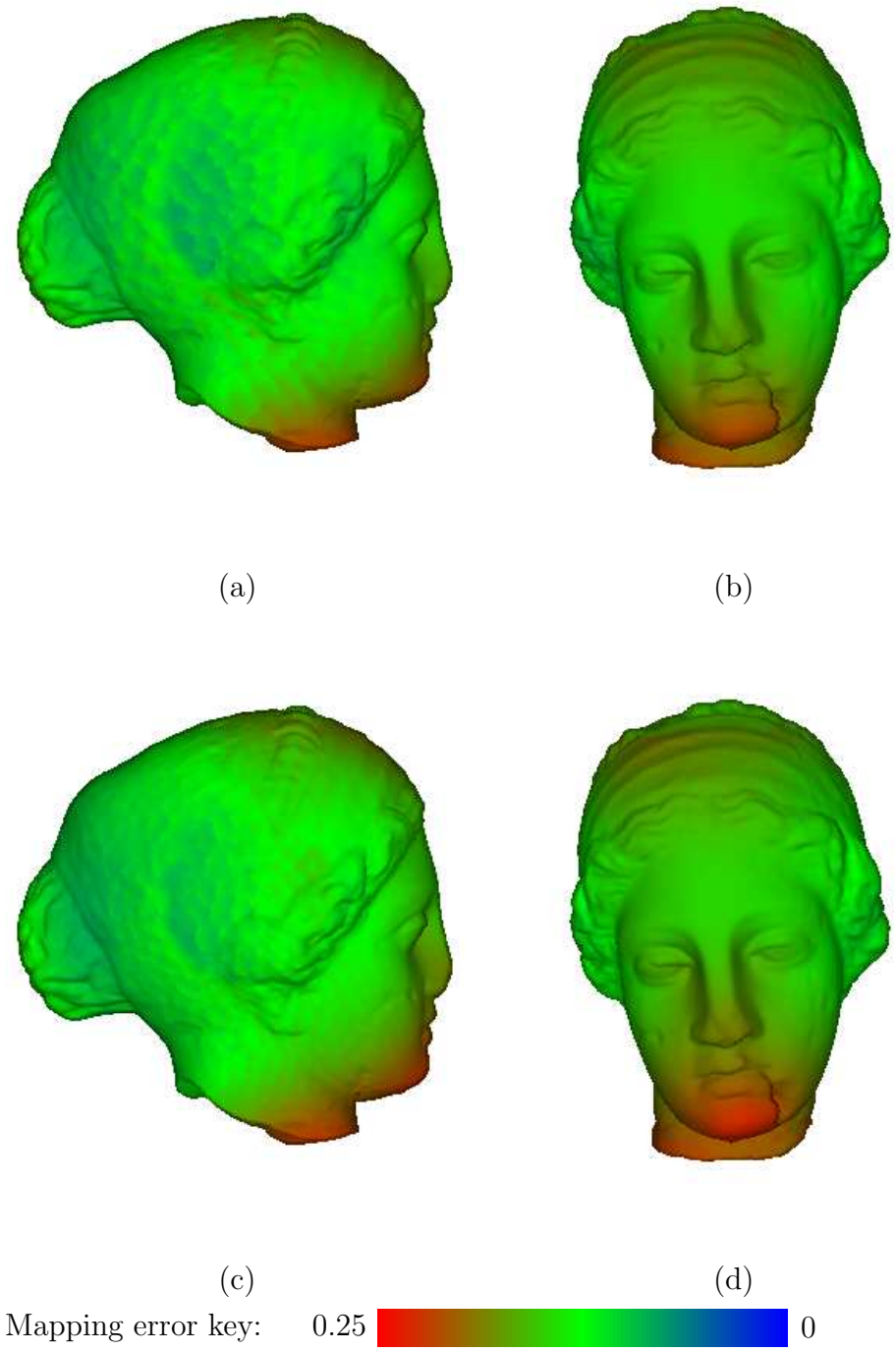


Fig. 4. Constrained spherical MDS [12] followed by interpolation. The mapping error of each surfel in the Igea model is color-coded. (a), (b): Using 1000 reference surfels. (c), (d): Using 2000 reference surfels.

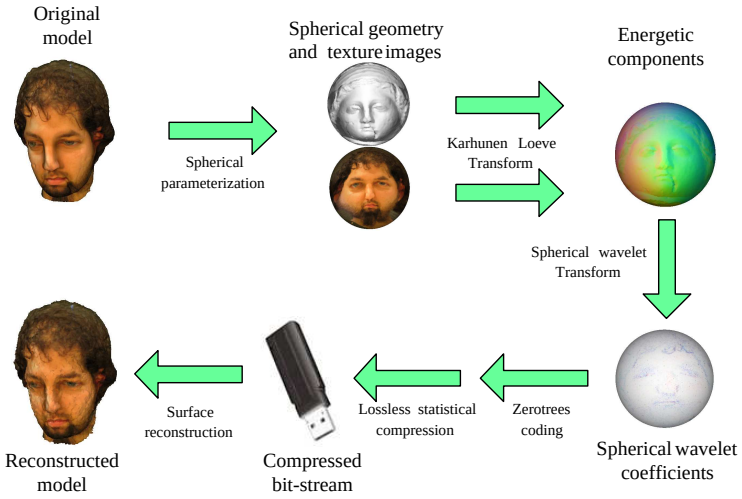


Fig. 5. The proposed compression pipeline.

4.1 Representation and compression scheme

In the suggested method for representation and compression of textured surfel-based surfaces, the underlying continuous surface is optimally mapped onto the unit sphere, as described in section 3. The mapping of each surfel includes its spatial coordinates and color components as a vector. Additional attributes, such as normal directions and surface reflectance properties can also be mapped. On the sphere, the mapped vectors are regarded as samples of a continuous vector valued (multilayered) function. This function is represented by spherical wavelets [38,39]. The wavelet coefficients are quantized, encoded and interleaved. The resulting bitstream is then losslessly compressed using an entropy encoder.

In the reconstruction stage, resurfeling takes place: a surfel is created for each spherical grid point. Its position and color attributes are obtained from the coded data. In the examples shown in this paper, the orientation and radius are derived from the spatial position using neighborhood considerations. The proposed compression pipeline is illustrated in Fig. 5. A compression result is shown in Fig. 6.

Hierarchical spherical grid

Consider the problem of determining a uniform sampling grid on a sphere. Dutton [40] obtained uniform spherical grids by tightly inscribing a regular polyhedron within the sphere, and taking the vertices that touch the sphere as the spherical grid points. Only for a tetrahedron, octahedron and icosahedron, the resulting spherical grid can be triangulated by identical equilateral

(a) (b)

Fig. 6. Compression of a textured Igea model (134345 surfels). (a) Original textured model. (b) Compression to 3.56 bits/surfel.

spherical triangles. This initial regular mesh can be recursively subdivided to obtain a hierarchical spherical sampling grid. As suggested by [38], in this paper an icosahedron-based grid subdivision scheme is used. The resampling is at the first available resolution greater than the original resolution, i.e., at the first subdivision level containing more grid points than the original number of surfels. Resampling requires interpolation of the available scattered function values at the spherical grid positions. This is accomplished using the method of Franke and Nielson [35], that is based on proximity-weighted neighbor averaging.

Spherical wavelets representation

Following the dense spherical mapping and the resampling of the surface, we end up with a multi-layered (vector-valued) attribute function sampled on the hierarchical spherical grid. It is characterized by spatial redundancy as well as by inter-layer redundancy.

To remove the inter-layer redundancy, the Karhunen Loève transform (KLT) is employed. The vector-valued samples of the attribute function corresponding to the given surface are used to compute the covariance matrix. Each vector-valued sample is represented using the Karhunen Loève basis vectors derived from the covariance matrix. Note that the dimension of the KLT is very low, equal to the number of layers, i.e., to the total number of surface attributes that are carried into the representation (three spatial coordinates, three color components, etc.). Thus, its computation is cheap, and the overhead due to

its inclusion in the compressed file is negligible.

To remove the spatial redundancy, each layer is represented by spherical wavelet decomposition via the lifting scheme. This technique was presented by Sweldens [39], and later adapted for functions on spherical domains by Schröder and Sweldens [38]. In the lifting scheme, the wavelet basis function is not formulated explicitly, but is implicitly 'lifted' from level to level, using a 'mother' wavelet function. Here the butterfly subdivision function [41] is used as a down sampling 'mother' function. A lifting stage creates a down-scaled version of the original function, and a set of wavelet coefficients that can be later used for reconstruction. The resulting wavelet coefficients are encoded using the zerotree algorithm [42], and losslessly stored using a statistical coder.

4.2 Special case: geometry only

The compression of purely geometric (textureless) surface data is of practical importance, and has received much greater attention than the more general problem considered in subsection 4.1, see [37,4] and references therein.

The compression scheme presented in this paper can be reduced to the case of pure geometry compression in a straightforward manner. Beyond its significant practical value, this exercise creates the possibility of quantitative comparison of our approach to related methods, for which performance data is available.

The geometry-only compression pipeline is shown in Fig. 7. The three geometry layers $\{X, Y, Z\}$ are resampled on the hierarchical grid, decorrelated using the Karhunen-Loève Transform and coded using the zerotree algorithm followed by statistical encoding.

Evaluation of geometry error

In order to evaluate the geometry error due to the compression-reconstruction process, an error measure is needed. In the case of triangle mesh surfaces, the Metro tool [6], that defines an RMS error between two mesh surfaces, is commonly used. For surfaces represented by surfel sets, we follow the error estimation algorithm introduced by Ochotta and Saupe [4]. The error is estimated as the RMS distance between the Moving Least Squares (MLS) representations [7] of the original and reconstructed surfaces.

For a given set of points P (the surfel centers in this case), that represents a surface S, Alexa *et al* [7] define a projection operator $\mathcal{P}_P(q)$ to project a nearby point q onto the surface. Given two point sets, P and $\hat{P}$, containing the centers of the surfels representing surfaces S and $\hat{S}$ respectively, the distance

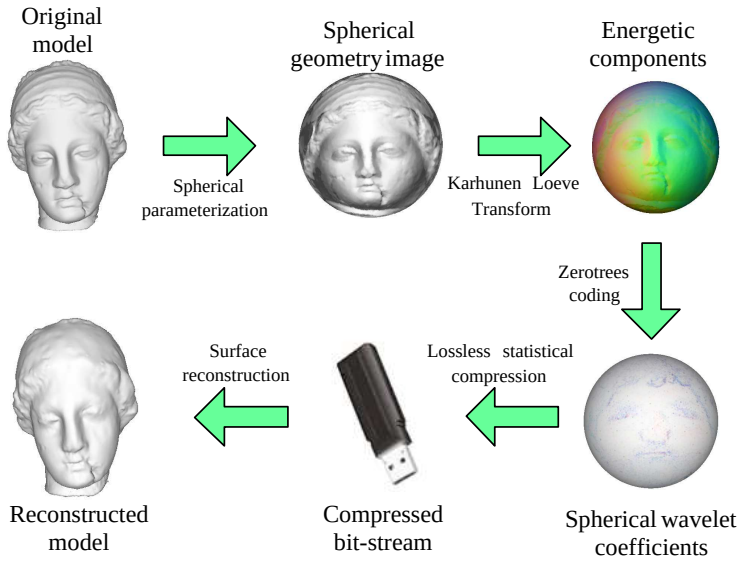


Fig. 7. Geometry only compression pipeline.

between a point $p \in P$ and the representation $\hat{P}$ of $\hat{S}$ is defined as

$$d(p, \hat{P}) = \|q - \mathcal{P}_{\hat{P}}(q)\| \tag{4}$$

where $q = \mathcal{P}_P(p)$ (see Fig 8), and the RMS error between S and $\hat{S}$ is

$$d(S, \hat{S}) = \left[\frac{1}{\|P\|} \sum_{p \in P} d^2(p, \hat{P}) \right]^{\frac{1}{2}} \tag{5}$$

Since this error measure is not symmetric, i.e., $d(\hat{S}, S) = d(S, \hat{S})$ is not generally true, a symmetric error is defined:

$$E_{RMS} = \max \left\{ d(\hat{S}, S), d(S, \hat{S}) \right\} \tag{6}$$

The RMS error is expressed in $\%d_{bb}$ units, where d_{bb} denotes the diagonal of the intersection of the (nearly identical) bounding boxes of the two surfaces [1]. The measure E_{RMS} quantifies the surface geometry error introduced by the compression process, and does not take texture into account. It allows to evaluate the dependence of the geometry distortion on the compression ratio in the proposed scheme, and enables comparison with other surfel based compression methods with respect to geometric distortion. A geometry error analysis example is presented in Fig. 9. Fig. 9c is the visualization of the local discrepancy between the outcome of the compression process (Fig. 9b) and the original surface (Fig. 9a). The global error in this example is $E_{RMS} = 3.01 \cdot 10^{-4}\ d_{bb}$.

[1] For a surface tightly bounded by the unit cube, $d_{bb} = \sqrt{3}$, and $\%d_{bb}$ denotes percents of that number.

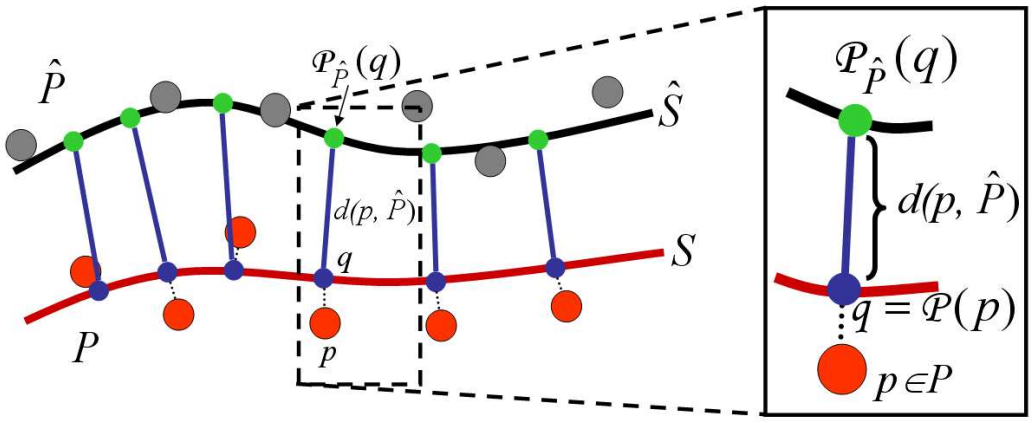


Fig. 8. Measuring the distance between two surfaces S (red curve) and $\hat{S}$ (black curve) represented by two point sets P (red dots) and $\hat{P}$ (grey dots), respectively. Each point $p \in P$ is projected to a point $q = \mathcal{P}_P(p) \in S$ (blue dot). The point $q \in S$ is then projected onto the surface $\hat{S}$, resulting in the point $\mathcal{P}_{\hat{P}}(q) \in \hat{S}$ (green dot). The distance $d(p, \hat{P})$ is the Euclidean distance between these two points, $\|q - \mathcal{P}_{\hat{P}}(q)\|$.

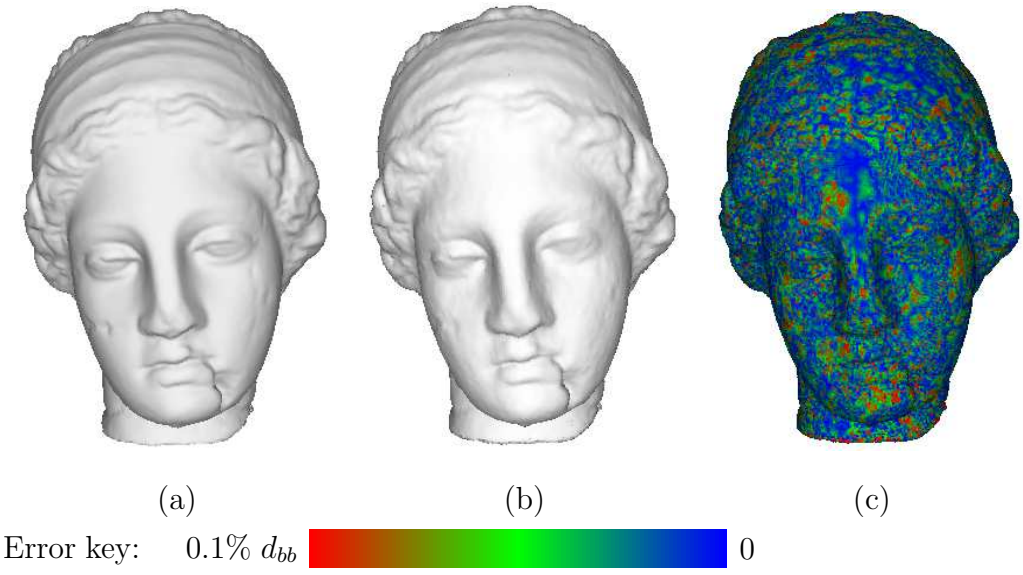


Fig. 9. Example of error measurement: (a) original *Igea* model. (b) Compressed model. (c) Error mapped on the compressed model, total $E_{RMS} = 3.01 \cdot 10^{-4}\ d_{bb}$

5 Results

5.1 Geometry only compression

Experimental results are presented for the *Igea* model, originally taken from the Cyberware 3D samples library and converted to surfel-based representation by the Pointshop3D team at ETH Zurich. Pointshop3D software [2] was also used to render the results throughout this paper. The texture-free 3D *Igea* model, consisting of 134345 surfels, technically at 96 bits/surfel, is shown in Fig. 10a. Fig. 10b–d show the outcome of the proposed compression scheme with 5.17 bits/surfel, 1.22 bits/surfel and 0.42 bits/surfel respectively. The corresponding point-wise reconstruction error $d(p, \hat{P})$ is visualized in Fig. 10e–g. The respective overall error values E_{RMS} are $3.48 \cdot 10^{-4}$ d_{bb}, $6.66 \cdot 10^{-4}$ d_{bb} and $9.99 \cdot 10^{-4}$ d_{bb}. Fig. 11 shows E_{RMS} as a function of the bit-rate and gives a quantitative comparison to the results of Ochotta and Saupe [4], Waschbüsch *et al* [5] and Fleishman *et al* [3]. The method presented in this paper is superior to the method of [4] at the lower bit-rates. The method of [3], that provides excellent results at higher bit-rates, is inapplicable at very low bit rates due to its need to encode a base model [4]. Waschbüsch *et al* [5] do not report results at very low bit-rates. Low bit-rate compression results for the *Max Planck* model [23] are presented in Fig. 12.

5.2 Compression of textured surfaces

Since the *Igea* model is provided without texture, it was mapped with a facial texture using Pointshop3D. The textured *Igea* model, technically with 120 bits/surfel, is shown in Fig. 13a. Fig. 13b is the result of compression of the original surface to 3.56 bits/surfel. Fig. 13c is the result of compression to 0.89 bits/surfel. An additional example of textured 3D model compression is presented in Fig. 14.

6 Future work: perceptually guided compression

In the compression scheme presented in subsection 4.1, geometry and texture values are treated equally. However, geometry and texture errors influence the visual quality of the rendered surface differently. Having the possibility to explicitly control the geometry and texture compression rates, using a perceptual criterion, can potentially lead to superior visual quality of the compression outcome.

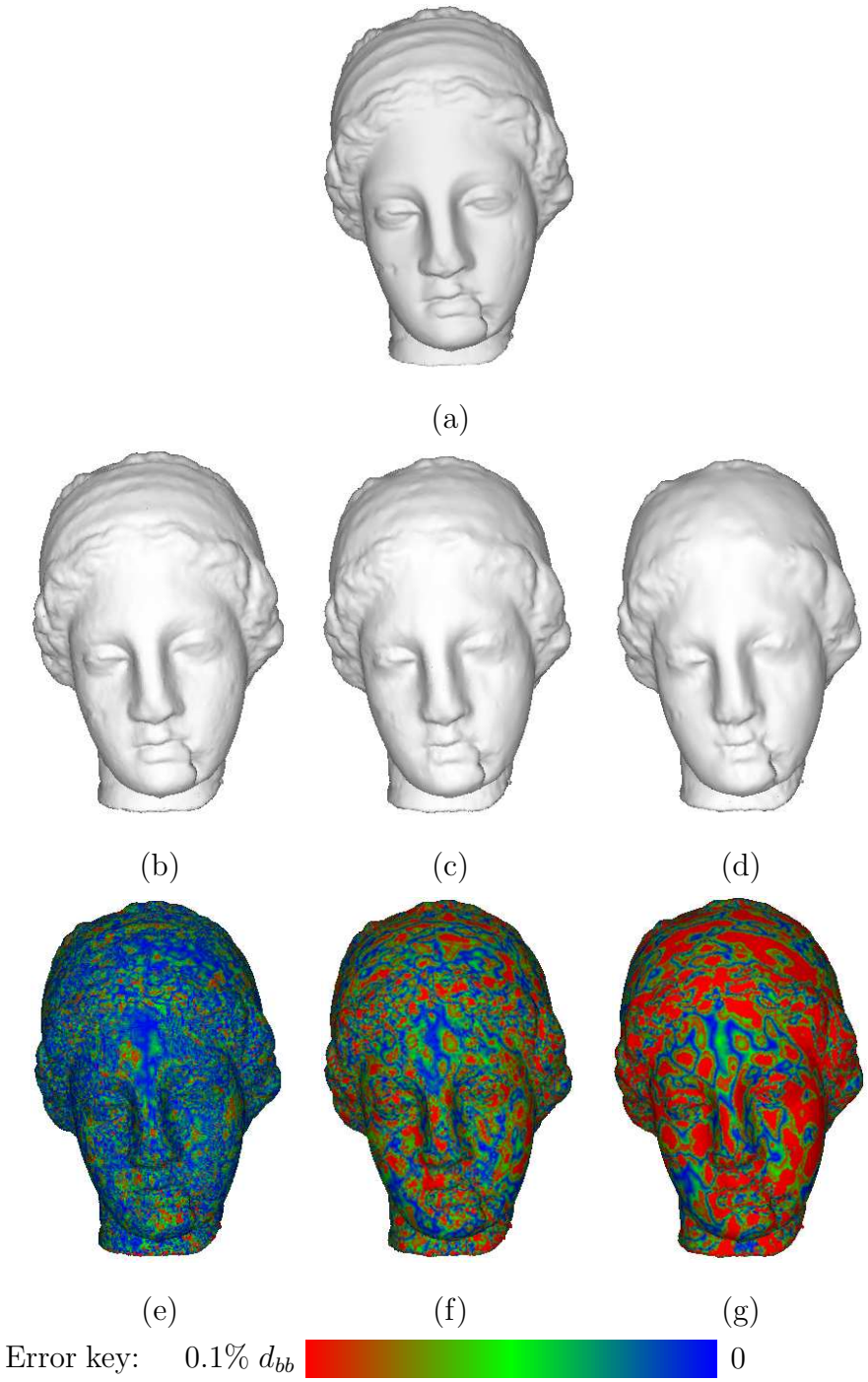


Fig. 10. Compression of the *Igea* model (geometry only). (a) Original model. (b) Compressed to 5.17 bits/surfel. (c) Compressed to 1.22 bits/surfel. (d) Compressed to 0.42 bits/surfel. (e) The point-wise reconstruction error in (b). (f) Error in (c). (g) Error in (d).

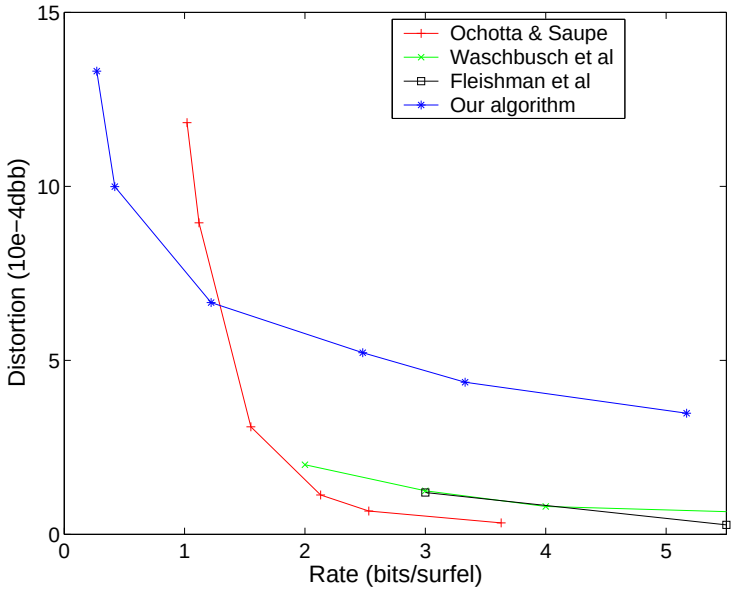
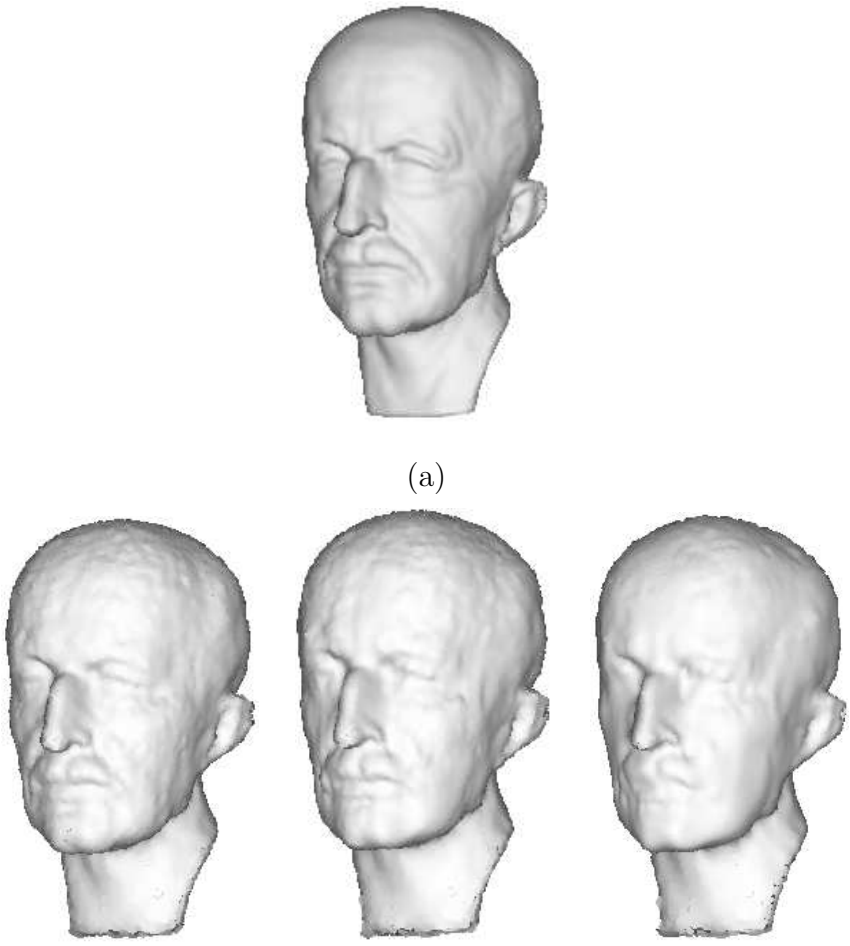


Fig. 11. A comparison of the geometry compression error E_{RMS} as a function of the bit-rate in the compression of the texture-free *Igea* model.

As described in subsection 4.1, following the Karhunen-Loève decorrelation step, the output channels can be associated with neither geometry nor texture. Here we discuss the possibility of modifying the algorithm to allow for perceptually-guided geometry-texture bit allocation.

In the modified algorithm, the wavelet coefficients representing the geometry and the texture are separately encoded, yielding two bitstreams. To resolve the bit allocation problem that arises in the subsequent interleaving step, we should seek a perceptual quality criterion, leading to the compression scheme illustrated in Fig. 15.

Horbelt *et al* [43] suggested a quantitative criterion, based on the L_2 error between rendered images of the three dimensional model, with and without compression. Nevertheless, the correspondence between standard error norms and visual quality is limited. Recently, Basu *et al* [44] studied the texture-geometry resource-allocation problem towards the goal of maximizing the perceived quality. Their research was in the context of 3D model simplification, in which geometry and texture were subsampled at different densities. Based on extensive psychophysical experiments, they derived an explicit mathematical expression that approximates the perceptual quality as a function of the geometry and texture detail levels. Even though the specific psychophysical results obtained by Basu *et al* [44] are not directly applicable in this research, their work demonstrates how psychophysical evaluation of the perceptual quality criterion can be carried out. Performing this psychophysical study is an interesting topic for future research.



(a)

(b) (c) (d)

Fig. 12. Low bit-rate compression of the *Max Planck* model (geometry only). (a) Original model. (b) Compressed to 3.50 bits/surfel, the error is $E_{RMS} = 14.1 \cdot 10^{-4}\ d_{bb}$. (c) Compressed to 1.98 bits/surfel, $E_{RMS} = 16.5 \cdot 10^{-4}\ d_{bb}$. (d) Compressed to 0.98 bits/surfel, $E_{RMS} = 18.4 \cdot 10^{-4}\ d_{bb}$.

To demonstrate the potential gains of perceptually-guided bit allocation, visually-appealing bit-allocation values were (manually) found for the compression of the textured *Igea* model at selected rates. Fig. 16a shows the original textured model. Compression of the *Igea* textured model with favorable geometry-texture bit allocation is presented in Fig. 16b, c and d at 0.48, 4.75 and 9.4 bits/surfel respectively. The compression results presented in Fig. 16e, f and g are at the same bit-rates, but with less than ideal bit-allocation. The value of perceptually-guided bit-allocation is noticeable especially at low bit-rates.

(a) (b) (c)

Fig. 13. Compression of the *Igea* 3D model with texture. (a) Original model, 134345 surfels, technically 120 bits/surfel. (b) Compression to 3.56 bits/surfel. (c) Compression to 0.89 bits/surfel.

(a) (b) (c)

Fig. 14. Compression of the textured *Trilobite* model (3D model courtesy of Arius3D Inc.). (a) Original model, 726027 surfels, technically 120 bits/surfel. (b) Compression to 6.3 bits/surfel. (c) Compression to 3.77 bits/surfel.

7 Discussion

We presented a surface compression scheme designed for genus-0 surfaces represented by surfel sets. It handles various surface attributes, notably geometry and texture, within a unified representation framework. The compression-induced geometry error is smaller than that of other methods at very low bit-rates.

The suggested compression scheme is ready to incorporate a perceived visual

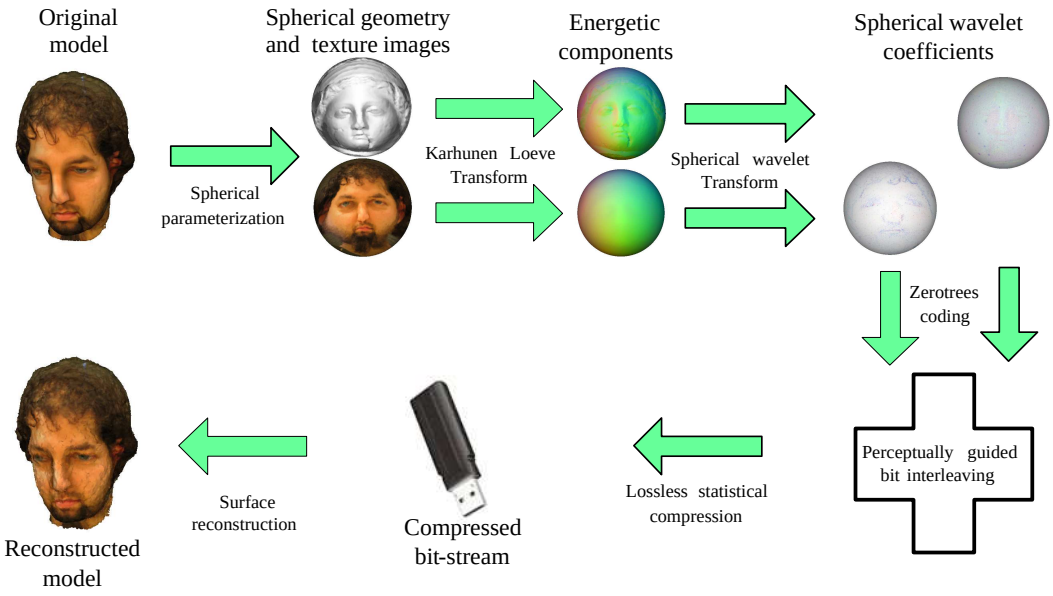


Fig. 15. Perceptual quality guided compression scheme.

quality criterion, that can be obtained using the methodology of [44]. This can lead to perceptually-optimal resolution of the geometry-texture bit-allocation tradeoff.

The proposed compression scheme necessitated the development of a geodesic path finder for surfel-based surfaces (section 2). Our approach for computing geodesics based on neighborhood graphs was motivated by the application of spherical parameterization, in which it is not crucial to have exact geodesic distances. Our method provides a good balance between performance and efficiency. Unlike shortest path algorithms for point-clouds, e.g. [16], the suggested method utilizes the normal direction and radius attributes available in the surfel representation. For higher precision one may introduce a relaxation algorithm that improves the piecewise linear geodesic approximation [45]. For this purpose we propose to apply an energy minimization taking into account Euclidean distance, closeness to the surface, and distribution of the sampling of the geodesics. However, such an approach must be feasible to compute many geodesics in a short time. One possibility would be to apply only few iterations of the Hofer and Pottmann [17] technique modified for surfels, in order to produce at least a slight improvement in the precision of our geodesic's estimates. Furthermore, one could apply Memoli and Sapiro's technique [16] on a sufficiently dense 3D grid to obtain more precise geodesics. In both these scenarios the tradeoff between geodesics accuracy, execution time and distortion of spherical parameterization plays a fundamental role.

A major contribution of this paper is a novel dense spherical mapping algorithm. Unlike previous approaches [12], there is no need to down-sample the

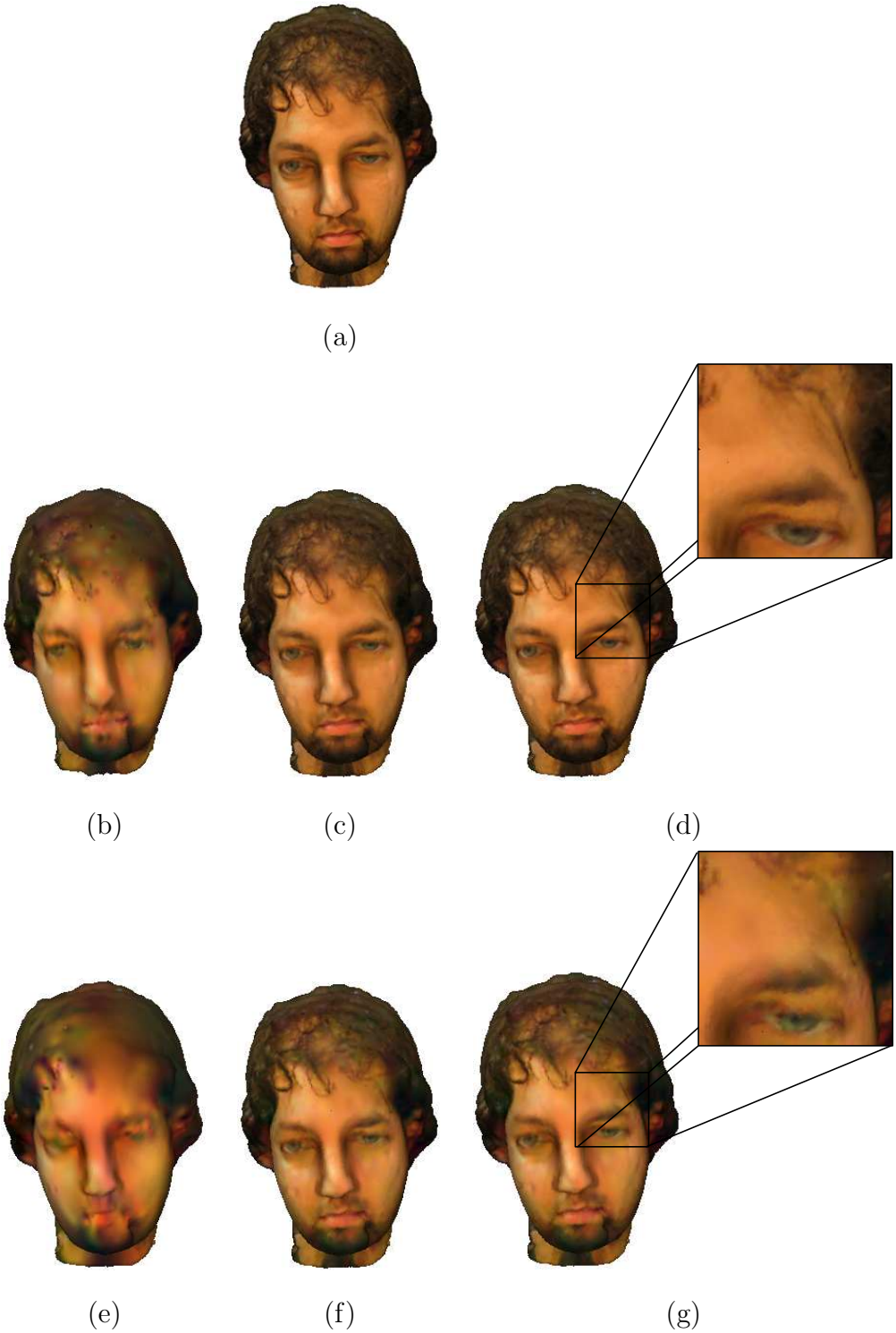


Fig. 16. The value of perceptually-guided bit-allocation. (a) Original textured model. (b,c,d) Compression with perceptually-sound bit-allocation, at 0.48, 4.75 and 9.4 bits/surfel respectively. (e,f,g) Compression with less than ideal bit-allocation, at 0.48, 4.75 and 9.4 bits/surfel.

surface or interpolate the solution. The introduction of rectangular (rather than square) distance matrices in MDS-type surface flattening dramatically lowers the memory needed for a given resolution and accuracy, and turns accurate spherical mapping of large surfaces into a computationally-feasible task. Following Bankirer [31], this approach is expected to have an impact on algorithms for surface flattening onto the plane [29,30].

References

[1] X. Gu, S. Gortler, H. Hoppe, Geometry images, in: Proceedings of SIGGRAPH, 2002, pp. 355–361.

[2] H. Pfister, M. Zwicker, J. van Baar, M.Gross, Surfels: surface elements as rendering primitives, in: Proceedings of SIGGRAPH, 2000, pp. 335–342.

[3] S. Fleishman, D. Cohen-Or, M. Alexa, C. T.Silva, Progressive point set surfaces, ACM Transactions on Graphics 22 (4) (2003) 997–1011.

[4] T. Ochotta, D. Saupe, Compression of point-based 3d models by shape-adaptive wavelet coding of multi-height fields, in: Proceedings of Symposium on Point-Based Graphics, Zürich, Switzerland, 2004.

[5] M. Waschbüsch, M. Gross, F. Eberhard, E. Lamboray, S. Würmlin, Progressive compression of point-sampled models, in: Proceedings of Symposium on Point-Based Graphics, 2004, pp. 95–102.

[6] P. Cignoni, C. Rocchini, R. Scopigno, Metro: measuring error on simplified surfaces, Computer Graphics Forum 17 (2) (1998) 167–174.

[7] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C. T. Silva, Computing and rendering point set surfaces, IEEE Transactions on Visualization and Computer Graphics 9 (1) (2003) 3–15.

[8] T. Cox, M. Cox, Multidimensional scaling on a sphere, Communications in Statistics: Theory and Methods 20 (1991) 2943–2953.

[9] A. Bronstein, M. Bronstein, R. Kimmel, Expression-invariant face recognition via spherical embedding, in: Proceedings of the IEEE International Conference on Image Processing, 2005, pp. 756–759.

[10] A. Elad, Y. Keller, R. Kimmel, Texture mapping via spherical multi-dimensional scaling, in: Proc. of Scale Space, 2005, pp. 443–455.

[11] T. Darom, M. Ruggeri, D. Saupe, N. Kiryati, Processing of textured surfaces represented as surfel sets: Representation, compression and geodesic paths, in: Proceedings of the IEEE International Conference on Image Processing, 2005, pp. 605–608.

[12] A. Elad, R. Kimmel, Spherical flattening of the cortex surface, in: R. Malladi (Ed.), Geometric Methods in Biomedical Image Processing, Springer, 2002.

[13] N. Kiryati, G. Székely, Estimating shortest paths and minimal distances on digitized three dimensional surfaces, Pattern Recognition 26 (1993) 1623–1637.

[14] R. Kimmel, J. Sethian, Computing geodesics on manifolds, Proceedings of the National Academy of Science 95 (1998) 8431–8435.

[15] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, H. Hoppe, Fast exact and approximate geodesics on meshes, ACM Trans. Graph. 24 (3) (2005) 553–560.

[16] F. Memoli, G. Sapiro, Distance functions and geodesics on point clouds, Tech. Rep. 1925, University of Minnesota (May 2003).

[17] M. Hofer, H. Pottmann, Energy-minimizing splines in manifolds, ACM Transactions on Graphics 23 (2004) 284–293.

[18] J. Klein, G. Zachmann, Proximity graphs for defining surfaces over point clouds, in: Symposium on Point-Based Graphics, ETH, Zürich, Switzerland, 2004.

[19] J. Sethian, A fast marching level-set method for monotonically advancing fronts, Proceedings of the National Academy of Science 93 (1996) 1591–1595.

[20] J. Jaromczyk, G. Toussaint, Relative neighborhood graphs and their relatives, in: Proceedings of the IEEE, Vol. 80, 1992, pp. 1502–1517.

[21] A. V. Goldberg, Shortest path algorithms: Engineering aspects, in: ISAAC '01: Proceedings of the 12th International Symposium on Algorithms and Computation, Springer-Verlag, London, UK, 2001, pp. 502–513.

[22] The stanford 3d scanning repository, `http://www-graphics.stanford.edu/data/3Dscanrep/`.

[23] Pointshop3d repository, `http://graphics.ethz.ch/pointshop3d/download.html`.

[24] The qsplat multiresolution point rendering system repository, `http://graphics.stanford.edu/software/qsplat/models/`.

[25] Konstanz 3d model repository, `http://www.inf.uni-konstanz.de/cgip/projects/surfac/`.

[26] E. Praun, H. Hoppe, Spherical parametrization and remeshing, ACM Transactions on Graphics 22 (2003) 340–349.

[27] H. Hoppe, E. Praun, Shape compression using spherical geometry images, in: N. Dodgson, M. Floater, M. Sabin (Eds.), Advances in Multiresolution for Geometry Modelling, Springer-Verlag, 2005, pp. 27–46.

[28] M. Zwicker, C. Gotsman, Meshing point clouds using spherical parameterization, in: Proceedings of Symposium on Point-Based Graphics, 2004.

[29] G. Zigelman, R. Kimmel, N. Kiryati, Texture mapping using surface flattening via multi-dimensional scaling, IEEE Transactions on Visualization and Computer Graphics 8 (2002) 198–207.

[30] R. Grossmann, N. Kiryati, R. Kimmel, Computational surface flattening: A voxel-based approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 433–441.

[31] C. Bankirer, N. Kiryati, Efficient surface flattening, Technical Report, in preparation.

[32] I. Borg, P. Groenen, Modern Multidimensional Scaling, Springer Series in Statistics, Springer, 1997.

[33] Y. Eldar, M. Lindenbaum, M. Porat, Y. Y. Zeevi, The furthest point strategy for progressive image sampling, IEEE Transactions on Image Processing 6 (9) (1997) 1305–1315.

[34] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical Recipies in C++. The Art of Scientific Computing, 2nd Edition, Cambridge University Press, 2002.

[35] R. Franke, G. Nielson, Smooth interpolation of large sets of scattered data, International Journal for Numerical Methods in Engineering 15 (1980) 1691–1704.

[36] C. Touma, C. Gotsman, Triangle mesh compression, in: Proceedings Graphics Interface, 1998, pp. 26–34.

[37] A. Khodakovsky, P. Schröder, W. Sweldens, Progressive geometry compression, in: Proceedings of SIGGRAPH, 2000, pp. 271–278.

[38] P. Schröder, W. Sweldens, Spherical wavelets: Efficiently representing functions on the sphere, in: Proceedings of SIGGRAPH, 1995, pp. 161–172.

[39] W. Sweldens, The lifting scheme: A construction of second generation wavelets, SIAM Journal on Mathematical Analysis 29 (2) (1998) 511–546.

[40] G. Dutton, Locational properties of quaternary triangular meshes, in: Proceedings of the 4th International Symposium on Spatial Data Handling, 1990, pp. 901–910.

[41] N. Dyn, D. Levin, J. Gregory, A butterfly subdivision scheme for surface interpolation with tension control, ACM Transactions on Graphics 9 (2) (1990) 160–169.

[42] J. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, IEEE Transactions on Signal Processing 41 (12) (1993) 3445–3462.

[43] S. Horbelt, L. Balmelli, M. Vetterli, Joint mesh and texture compression using marginal analysis, in: Proceedings of the IEEE International Conference on Image Processing, Vol. 3, 2001, pp. 114–117.

[44] A. Basu, I. Cheng, T. Wang, Balanced incomplete designs for 3D perceptual quality estimation, in: Proceedings of the IEEE International Conference on Image Processing, 2005.

[45] M. Ruggeri, T. Darom, D. Saupe, N. Kiryati, Approximating geodesics on point set surfaces, in: Proceedings of Symposium on Point-Based Graphics, Boston, Massachusetts, 2006.